Statistical Significance and Performance Measures

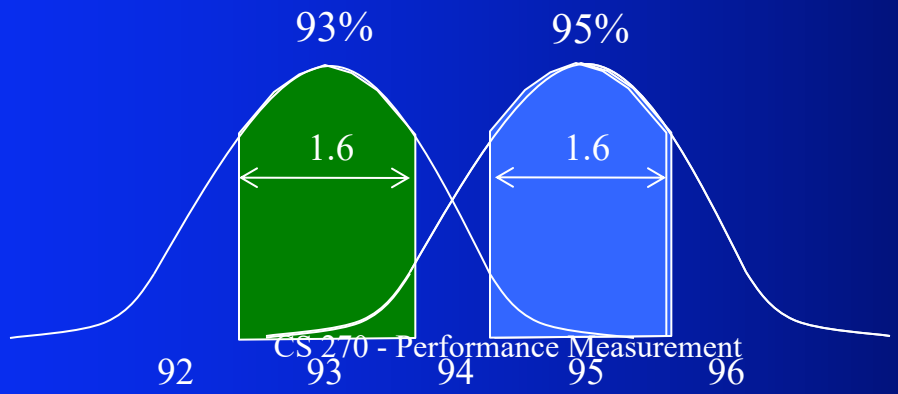
- Just a brief review of confidence intervals since you had these in Stats – Assume you've seen *t*-tests, etc.
  - Confidence Intervals
  - Statistical Significance
- Permutation Testing
- Other Performance Measures
  - Precision
  - Recall
  - F-score
  - ROC

Statistical Significance

- How do we know that some measurement is statistically significant vs being just a random perturbation
  - How good a predictor of generalization accuracy is the sample accuracy on a test set?
  - Is a particular hypothesis really better than another one because its accuracy is higher on a validation set?
  - When can we say that one learning algorithm is better than another for a particular task or set of tasks?
- For example, if learning algorithm 1 gets 95% accuracy and learning algorithm 2 gets 93% on a task, can we say with some confidence that algorithm 1 is superior in general for that task?
- Question becomes: What is the likely difference between the sample error (estimator of the parameter) and the true error (true parameter value)?
- Key point – What is the probability that the differences in our results are just due to chance?

Confidence Intervals

- An N% confidence interval for a parameter p is an interval that is expected with probability N% to contain p
- The true mean (or whatever parameter we are estimating) will fall in the interval $\pm C_N\sigma$ of the sample mean with N% confidence, where σ is the deviation and C_N gives the width of the interval about the mean that includes N% of the total probability under the particular probability distribution. C_N is a distribution specific constant for different interval widths.
- Assume the filled-in intervals below are the 90% confidence intervals for our two algorithms. What does this mean?
  - The situation below says that these two algorithms are different with 90% confidence
  - Would if they overlapped?
  - How do you tighten the confidence intervals? – More data and tests



Central Limit Theorem

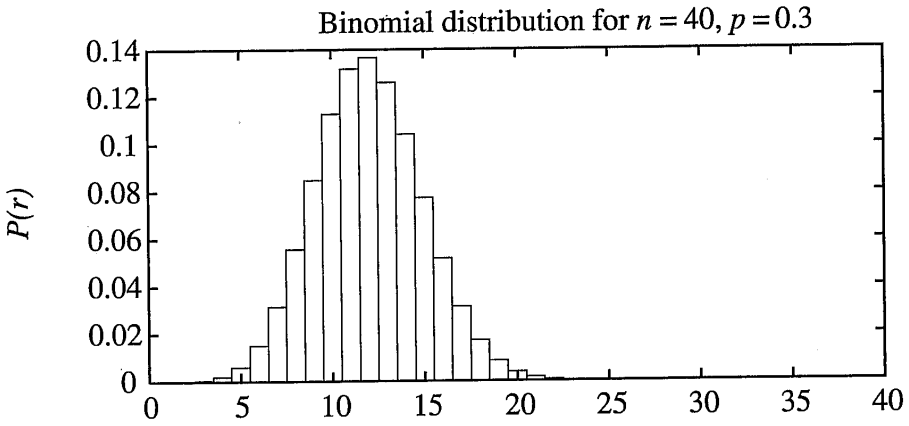
- Central Limit Theorem
  - If there are a sufficient number of samples, and
  - The samples are iid (independent, identically distributed) - drawn independently from the identical distribution
  - Then, the random variable can be represented by a Gaussian distribution with the sample mean and variance
- Thus, regardless of the underlying distribution (even when unknown), if we have enough data then we can assume that the estimator is Gaussian distributed
- And we can use the Gaussian interval tables to get intervals $\pm z_N \sigma$
- Note that while the test sets are independent in *n*-way CV, the training sets are not since they overlap (Still a decent approximation)

Binomial Distribution

- Given a coin with probability p of heads, the binomial distribution gives the probability of seeing exactly r heads in n flips.

$$P(r) = \frac{n!}{r!(n-r)!} p^r (1-p)^{n-r}$$

- A random variable is a random event that has a specific outcome (X = number of times heads comes up in n flips)
  - For binomial, $\Pr(X = r)$ is $P(r)$
  - The mean (expected value) for the binomial is np
  - The variance for the binomial is $np(1-p)$
- Same setup for classification where the outcome of an instance is either correct or in error and the sample error rate is r/n which is an estimator of the true error rate p



A *Binomial distribution* gives the probability of observing r heads in a sample of n independent coin tosses, when the probability of heads on a single coin toss is p. It is defined by the probability function

$$P(r) = \frac{n!}{r!(n-r)!} p^r (1-p)^{n-r}$$

If the random variable X follows a Binomial distribution, then:

- The probability $\Pr(X = r)$ that X will take on the value r is given by $P(r)$
- The expected, or mean value of X, $E[X]$, is

$$E[X] = np$$

- The variance of X, $Var(X)$, is

$$Var(X) = np(1-p)$$

- The standard deviation of X, σ_X, is

$$\sigma_X = \sqrt{np(1-p)}$$

For sufficiently large values of n the Binomial distribution is closely approximated by a Normal distribution (see Table 5.4) with the same mean and variance. Most statisticians recommend using the Normal approximation only when $np(1-p) \geq 5$.

Binomial Estimators

- Usually want to figure out p (e.g. the true error rate)
- For the binomial the sample error r/n is an unbiased estimator of the true error p
  - An estimator X of parameter y is unbiased if $E[X] - E[y] = 0$
- For the binomial the sample deviation is

$$\sigma_{err} = \frac{\sigma_r}{n} = \sqrt{\frac{np(1-p)}{n^2}} = \sqrt{\frac{p(1-p)}{n}} \approx \sqrt{\frac{Err_{sample}(1-Err_{sample})}{n}}$$

Comparing two Algorithms - paired *t* test

- Do k-way CV for both algorithms on the same data set using the same splits for both algorithms (paired)
  - Best if $k > 30$ but that will increase variance for smaller data sets
- Calculate the accuracy difference δ_i between the algorithms for each split (paired) and average the k differences to get δ
- Real difference is with N% confidence in the interval

$$\delta \pm t_{N,k\text{-}1}\ \sigma$$

where σ is the standard deviation and $t_{N,k\text{-}1}$ is the N% t value for k-1 degrees of freedom. The t distribution is slightly flatter than the Gaussian and the t value converges to the Gaussian (z value) as k grows.

Paired *t* test - Continued

- σ for this case is defined as $$\sigma = \sqrt{\frac{1}{k(k-1)} \sum_{i=1}^{k} (\delta_i - \delta)^2}$$
- Assume a case with $\delta = 2$ and two algorithms M_1 and M_2 with an accuracy average of approximately 96% and 94% respectively and assume that $t_{90,29} \times \sigma = 1$. This says that with 90% confidence the true difference between the two algorithms is between 1 and 3 percent. This approximately implies that the extreme averages between the algorithm accuracies are 94.5/95.5 and 93.5/96.5. Thus we can say that with 90% confidence that M_1 is better than M_2 for this task. If $t_{90,29} \times \sigma$ is greater than δ then we could not say that M_1 is better than M_2 with 90% confidence for this task.
- Since the difference falls in the interval $\delta \pm t_{N,k\text{-}1}\sigma$ we can find the $t_{N,k\text{-}1}$ equal to δ/σ to obtain the best confidence value

1. Partition the available data D_0 into k disjoint subsets $T_1, T_2, \ldots, T_k$ of equal size, where this size is at least 30.
2. For i from 1 to k, do

   use T_i for the test set, and the remaining data for training set S_i

   - $S_i \leftarrow \{D_0 - T_i\}$
   - $h_A \leftarrow L_A(S_i)$
   - $h_B \leftarrow L_B(S_i)$
   - $\delta_i \leftarrow error_{T_i}(h_A) - error_{T_i}(h_B)$
3. Return the value $\bar{\delta}$, where

$$\bar{\delta} \equiv \frac{1}{k} \sum_{i=1}^{k} \delta_i \tag{T5.1}$$

TABLE 5.5
A procedure to estimate the difference in error between two learning methods L_A and L_B. Approximate confidence intervals for this estimate are given in the text.

The approximate $N\%$ confidence interval for estimating the quantity in Equation (5.16) using $\bar{\delta}$ is given by

$$\bar{\delta} \pm t_{N,k-1}\, s_{\bar{\delta}} \tag{5.17}$$

where $t_{N,k-1}$ is a constant that plays a role analogous to that of z_N in our earlier confidence interval expressions, and where $s_{\bar{\delta}}$ is an estimate of the standard deviation of the distribution governing $\bar{\delta}$. In particular, $s_{\bar{\delta}}$ is defined as

$$s_{\bar{\delta}} \equiv \sqrt{\frac{1}{k(k-1)} \sum_{i=1}^{k} (\delta_i - \bar{\delta})^2} \tag{5.18}$$

Permutation Test

- With faster computing it is often reasonable to do a direct permutation test to get a more accurate confidence, especially with the common 10 fold cross validation (only 1000 permutations)

Menke, J., and Martinez, T. R., Using Permutations Instead of Student's *t* Distribution for *p*-values in Paired-Difference Algorithm Comparisons, Proceedings of the *IEEE International Joint Conference on Neural Networks IJCNN'04,* pp. 1331-1336, 2004.

- Even if two algorithms were really the same in accuracy you would expect some random difference in outcomes based on data splits, etc.
- How do you know that the measured difference between two situations is not just random variance?
- If it were just random, the average of many random permutations of results would give about the same difference (i.e. just the task variance)

Permutation Test Details

- To compare the performance of models M_1 and M_2 using a permutation test:

1. Obtain a set of k estimates of accuracy $A = \{a_1, ..., a_k\}$ for M_1 and $B = \{b_1, ..., b_k\}$ for M_2 (e.g. each do k-fold CV on the same task, or accuracies on k different tasks, etc.)
2. Calculate the average accuracies, $\mu_A = (a_1 + ... + a_k)/k$ and $\mu_B = (b_1 + ... + b_k)/k$ (note they are not paired in this algorithm)
3. Calculate $d_{AB} = |\mu_A - \mu_B|$
4. let $p = 0$
5. Repeat n times (or just every permutation)
   a. let $S=\{a_1, ..., a_k, b_1, ..., b_k\}$
   b. randomly partition S into two equal sized sets, R and T (statistically best if partitions not repeated)
   c. Calculate the average accuracies, μ_R and μ_T
   d. Calculate $d_{RT} = |\mu_R - \mu_T|$
   e. if $d_{RT} \geq d_{AB}$ then $p = p+1$
6. p-value = p/n (Report p, n, and p-value)

A low p-value implies that the algorithms really are different

	Alg 1	Alg 2	Diff
Test 1	92	90	2
Test 2	90	90	0
Test 3	91	92	-1
Test 4	93	90	3
Test 5	91	89	2
Ave	91.4	90.2	1.2

Statistical Significance Summary

- Required for publications
- No single accepted approach
- Many subtleties and approximations in each approach
  - Independence assumptions often violated
  - Degrees of freedom: Is LA_1 still better than LA_2 when
    - The size of the training sets are changed
    - Trained for different lengths of time
    - Different learning parameters are used
    - Different approaches to data normalization, features, etc.
    - Etc.
- Author's tuned parameters vs default parameters (grain of salt on results)
- Still can (and should) get higher confidence in your assertions with the use of statistical significance measures

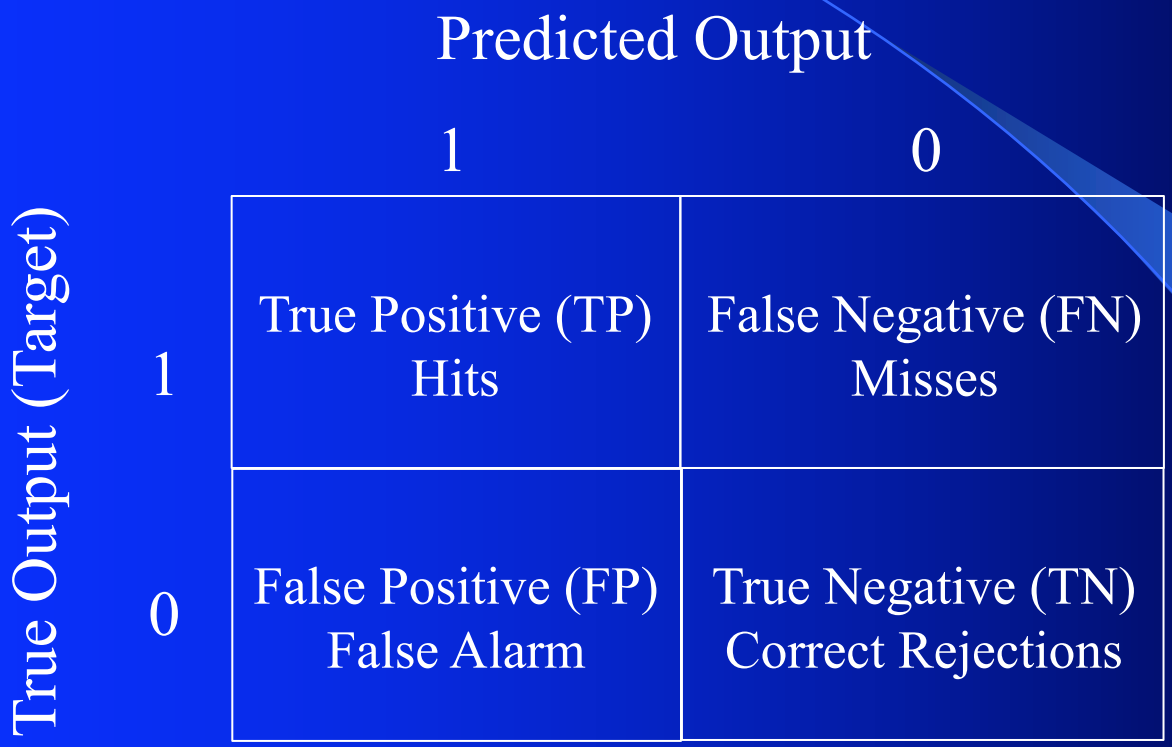
Performance Measures

- Most common measure is accuracy
  - Summed squared error
  - Mean squared error
  - Classification accuracy

Issues with Accuracy

- Is 99% accuracy good; Is 30% accuracy bad?
  - Depends on baseline and problem complexity
- Error reduction (1-accuracy)
  - Absolute vs relative
  - 99.90% accuracy to 99.99% accuracy is a 90% relative reduction in error, but absolute error is only reduced by .09%.
  - 50% accuracy to 75% accuracy is a 50% relative reduction in error and the absolute error reduction is 25%.
  - Which is better?
- Above assumes equal cost for all errors
  - Often have different error costs – e.g. Heart attack or not

Binary Classification



Accuracy = (TP+TN)/(TP+TN+FP+FN)

Precision = TP/(TP+FP)

Recall = TP/(TP+FN)

Recall

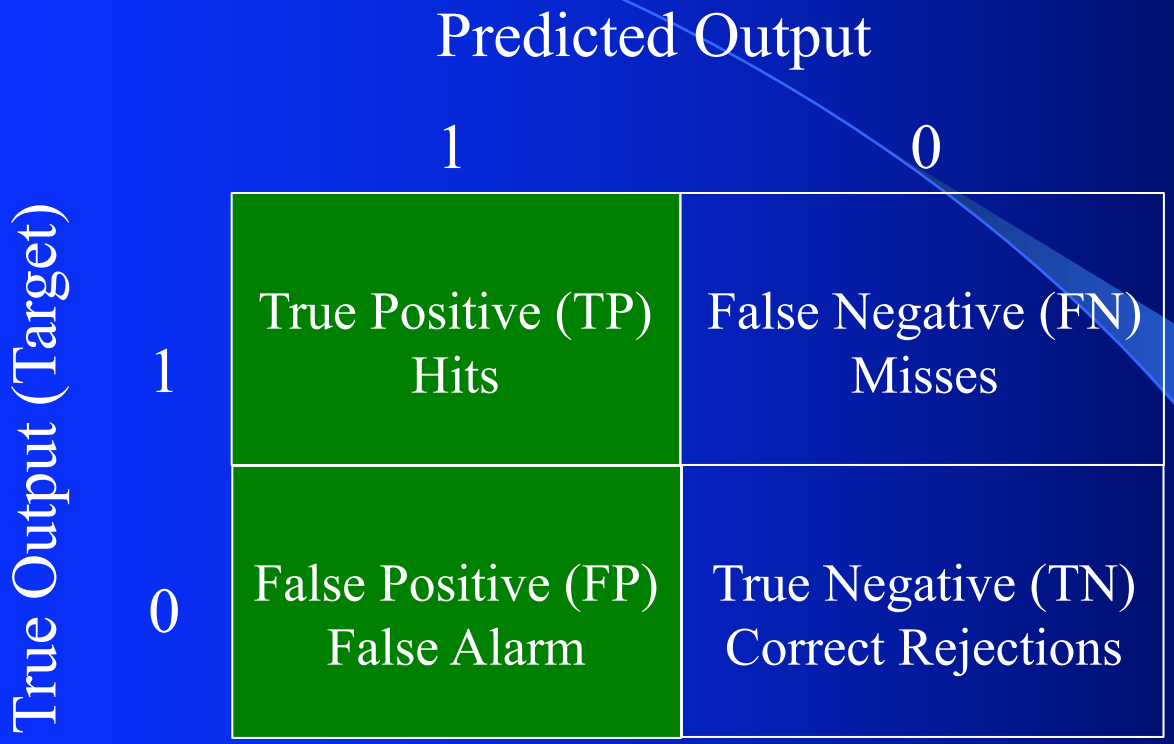
True Output (Target) \ Predicted Output	1	0
1	True Positive (TP) Hits	False Negative (FN) Misses
0	False Positive (FP) False Alarm	True Negative (TN) Correct Rejections

Recall = TP/(TP+FN)

The percentage of target true positives that were predicted as true positives, minimize false negatives

How to maximize?

Precision



Precision = TP/(TP+FP)

The percentage of predicted true positives that are target true positives, minimize false positives

How to maximize?

Other measures - Precision vs. Recall

- Find appropriate balance of Precision vs Recall for the task at hand, rather than just accuracy
- Can adjust ML parameters to accomplish the Precision vs Recall balance – Heart attack vs Google search
- Break even point: precision = recall
- F_1 or F-score = $2 \times (\text{precision} \times \text{recall}) / (\text{precision} + \text{recall})$ - Harmonic average of precision and recall
- One especially useful situation is when there is highly skewed data output, where accuracy may be misleading

Cost Ratio

- For binary classification (concepts) can have an adjustable threshold for deciding what is a True class vs a False class
  - For MLP it could be what threshold activation value is used to decide if a final output is true or false (default .5)
    - Could use .8 to get high precision or .3 for higher recall
  - For ID3 it could be what percentage of the leaf elements need to be in a class for that class to be chosen (default is the most common class)
- Could slide that threshold depending on your preference for True vs False classes (Precision vs Recall)
- Could measure the quality of an ML algorithm based on how well it can support this sliding of the threshold to dynamically support precision vs recall for different tasks - ROC

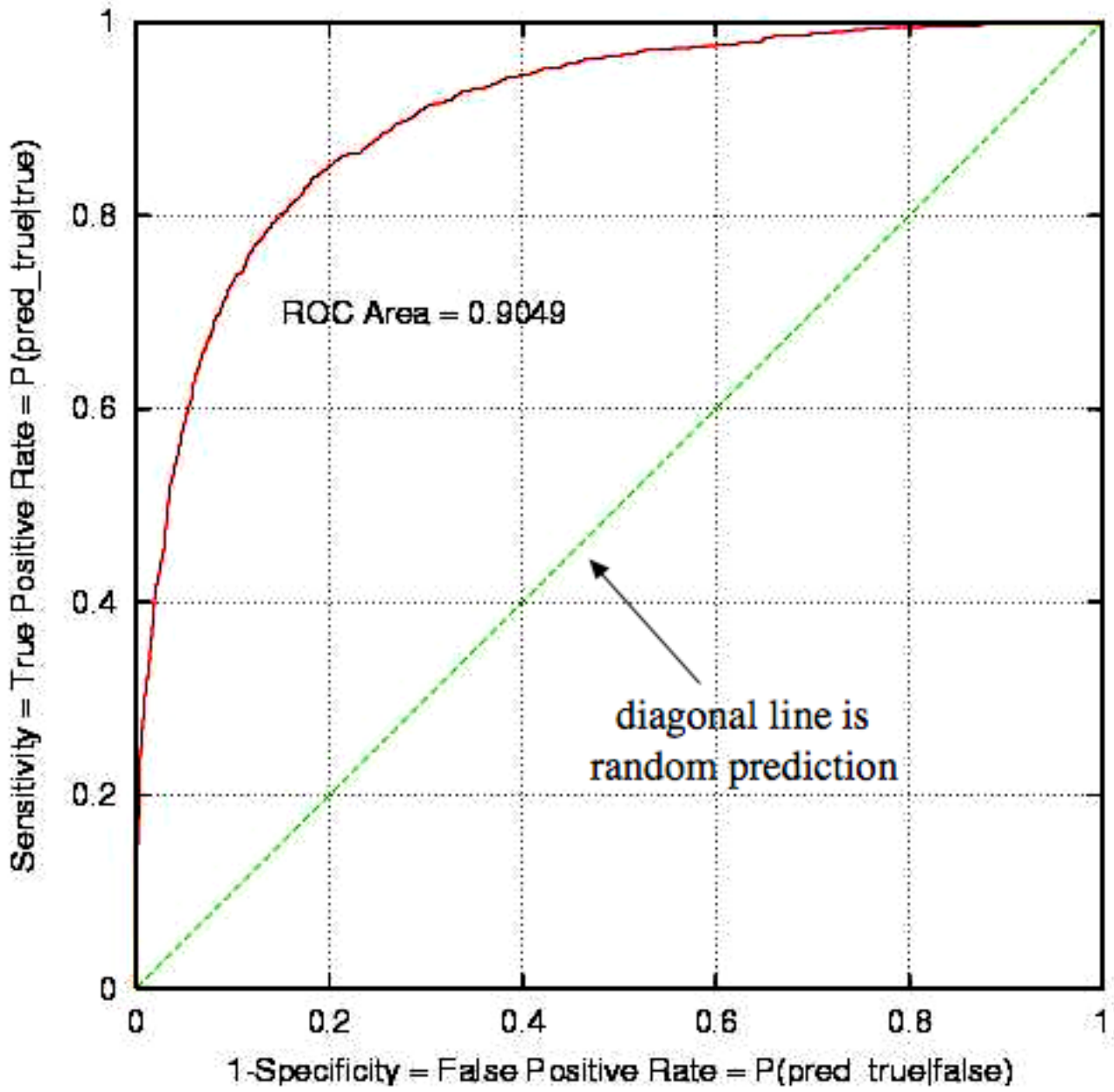
ROC Curves and ROC Area

- Receiver Operating Characteristic
- Developed in WWII to statistically model false positive and false negative detections of radar operators
- Standard measure in medicine and biology
- True positive rate (sensitivity) vs false positive rate (1- specificity)
- True positive rate (Probability of predicting true when it is true) P(Pred:T|T) = Sensitivity = Recall = TP/P = TP/(TP+FN)
- False positive rate (Probability of predicting true when it is false)

  P(Pred:T|F) = FP/N = FP/(TN+FP) = 1 – specificity (true negative rate) = 1 – TN/N = 1 - TN/(TN+FP)
  - Want to maximize TPR and minimize FPR
  - How would you do each independently?

ROC Curves and ROC Area

- Neither extreme is acceptable
  - Want to find the right balance
  - But the right balance/threshold can differ for each task considered
- How do we know which algorithms are robust and accurate across many different thresholds? – ROC curve
- Each point on the ROC curve represents a different tradeoff (cost ratio) between true positive rate and false positive rate
- Standard measures just show accuracy for one setting of the cost/ratio threshold, whereas the ROC *curve* shows accuracy for all settings and thus allows us to compare how robust to different thresholds one algorithm is compared to another


ROC Area = 0.9049
diagonal line is
random prediction
Sensitivity = True Positive Rate = P(pred_true|true)
1-Specificity = False Positive Rate = P(pred_true|false)
0
0.2
0.4
0.6
0.8
1

- Assume Backprop threshold
- Threshold = 1 (0,0), then all outputs are 0

  TPR = P(T|T) = 0

  FPR = P (T|F) = 0
- Threshold = 0, (1,1)

  TPR = 1, FPR = 1
- Threshold = .8 (.2,.2)

  TPR = .38 FPR = .02

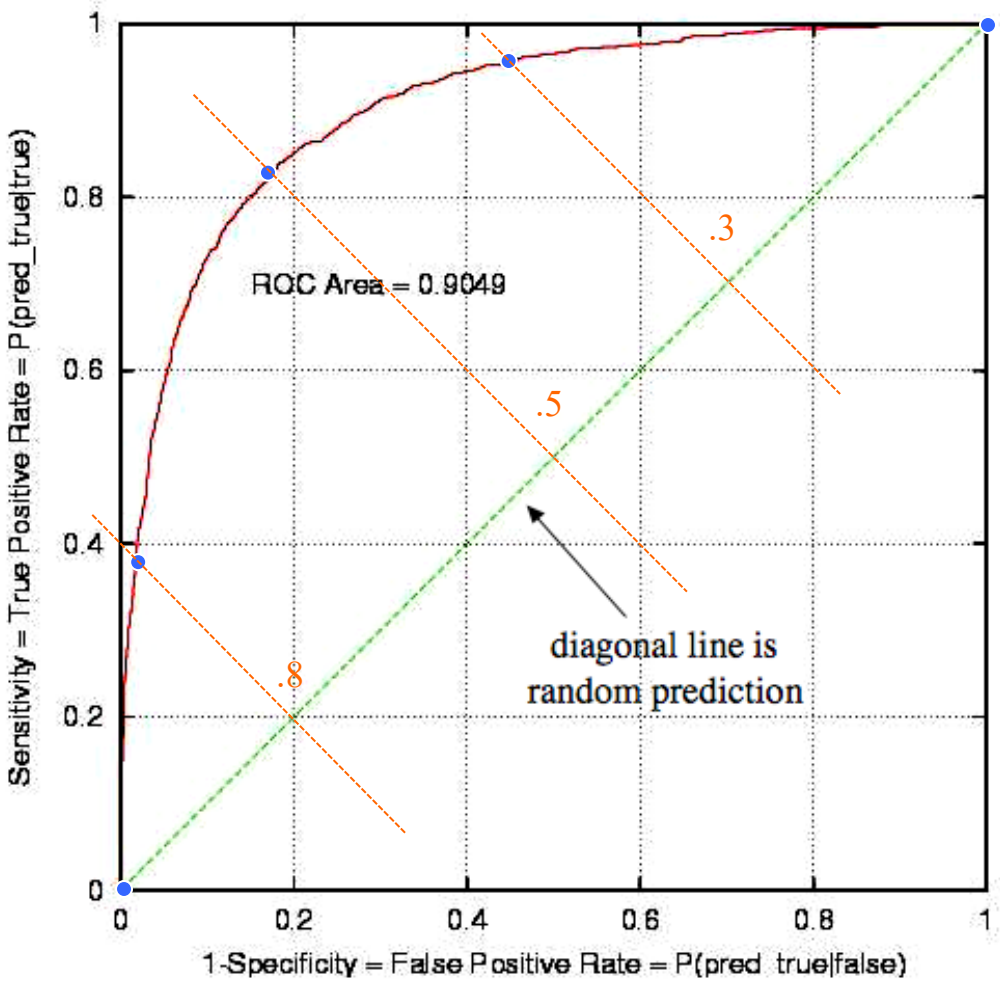
  - Good Precision, but recall (TPR) is low
- Threshold = .5 (.5,.5)

  TPR = .82 FPR = .18

  - Better Accuracy/balance
- Threshold = .3 (.7,.7)

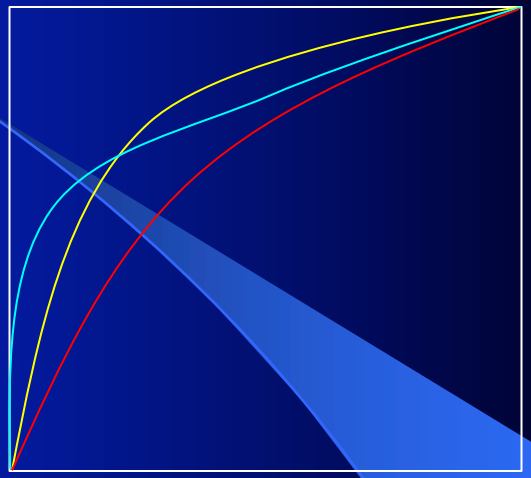
  TPR = .95 FPR = .43

  - Better Recall, worse precision



Accuracy is maximized at point closest to the top left corner. Note that Sensitivity = Recall and the lower the false positive rate, the higher the precision.

ROC Properties



- Area Properties
  - 1.0 - Perfect prediction
  - .9 - Excellent
  - .7 - Mediocre
  - .5 - Random
- ROC area represents performance over all possible cost ratios
- If two ROC curves do not intersect then one method dominates over the other
- If they do intersect then one method is better for some cost ratios, and is worse for others
  - Blue alg better for precision, yellow alg for recall, red neither
- Can choose method and balance based on goals

Performance Measurement Summary

- Other measures (e.g. Precision vs Recall, ROC, F-score) gaining popularity
- There are extensions to multi-output cases
  - However, medicine, finance, etc. have lots of two class problems
- Accuracy handles multi-class outputs and is still the most common measure but often combined with other measures like those above